MPS/CO Computer 71-2
19 April, 1971

PROPOSAL FOR THE IMLAC PDS1 SOFTWARE SYSTEM

H. van der Beken, B. Carpenter,
S. Jupp, W. Remmer

Introduction

Outline proposals for the PDS1 software were given in a previous note (MPS/CO Computer 71-1). This note gives more details of the general software to be implemented for communication between the IBM 1800 and the mini-computers, and for display generation in the mini-computers. The programs are now being written on the basis of these proposals and it is hoped that they will be operational in late summer.

The general software will consist of four main parts:

1) assembler program for PDS1 assembly language

2) programs and routines to use the disc file on which PDS1 coreloads will be stored

3) communications routines in the IBM

4) the 1 K EXECutive in the PDS1's.

Documentation will be organized by E. Ratcliff and will consist of three main parts:

a) assembler manual

b) programmer's guide to EXEC and communications procedures

c) subroutine descriptions.

A few copies of the manufacturer's manuals will be available for borrowing. In general only the hardware aspects of these are relevant.

/je

Assembler program

The assembler will run in the V-core of the IBM as a background-processing job. It will read in a program punched on cards in a language based on the manufacturer's assembly language. PDS1 object code will be created and the program can be listed on the line printer. If no errors occur during assembly, it will be possible to build the object code into a PDS1 coreload (PCL) and store it on the PCL disc file for subsequent execution.

Programs will be assembled absolutely and will not be relocatable. There will be no CALL facility for subroutines, which must be included as cards unless they are in the EXEC. There will be no LOCAL facility but the user may organize his own overlay if required.

It will not be possible to test a program unless it has first been stored on the PCL file.

Disc file

The PDS1's have been provisionally assigned 80 sectors of disc space, most of which will be available for PCL storage. Programs will be written to create the PCL file and to list its contents. Routines will be written to delete a particular PCL, to write a new PCL into the file and to read a PCL from the file into core.

Space in the file is restricted; there is room, for example, for approximately 23 PCL of 1 K each.

IBM communications routines

Each PDS1 will be able to send an interrupt bit to the IBM. When this occurs, an ISSR will read a 16 bit status word in the PDS1 interface. Fig. 1 shows the bits so far assigned in this status word.

15	PIV	
14	PRQ	Cause interrupt
13	RCS	
12	TCP	
11	READY	
10	ON-LINE	

Fig. 1

PIV PDS1 Interface Violation. This bit indicates that the software has attempted to transmit data in the wrong direction

PRQ PDS1 Requests. The PDS1 wishes to transmit to the IBM

RCS Requests Cold Start. The PDS1 requests the IBM to transmit its cold start program

TCP Transmission Complete

When the PIV bit is detected, the ISSR in the IBM should attempt a PDS1 restart. If this fails it should try a reload, and if this fails it should switch to the back-up PDS1. The next call to the transmission subroutines (see below) would return an error code.

When the PRQ bit is detected, the ISSR will read in one word of data from the PDS1. The top 8 bits of this will be a control function and the bottom 8 bits will normally be a word count. If the control function specifies "queue coreload" the IBM coreload specified in the queue indicator (see below) will be queued. If it specifies "abort" appropriate action will be taken. Otherwise the control function and word count will be passed over to the coreload using the PDS1 via the interrupt indicator (see below). In this case the control function will specify what type of file is waiting in the PDS1 for transmission to the partition (see Appendix).

Two skeleton flags are needed:

SKF1 : Interrupt indicator
Zero when no PRQ received
Non-zero when PRQ received and action is required by partition. Actually contains a control function and word count

SKF2 : Queue indicator.

Routines will be provided for transmitting files between a partition and a PDS1. The probable set of routines is:

1) read a specified PCL from disc, transmit to a PDS1

2) transmit a user-provided file to a PDS1. Useful file length up to approximately 240 words. The user must supply a control function and word count. See Appendix for control functions.

   (N.B. This routine will replace or duplicate CODDW, CPLOC, CPLOP, etc. Multiple entries may be provided if this seems more convenient.)

3) receive a file from a PDS1. Useful length up to approximately 240 words. Control function and word count were provided in SKF2; user must decode control function.

These routines and the ISSR will all be in the 1800 Executive and will share a word indicating the busy/free and OK/broken status of each PDS1, so that back-up operations wil be carried out automatically.

PDS1 EXECutive

The resident software in the PDS1 will consist of approximately 1 K of code and tables, leaving another 1 K for the USER program. The EXEC will be capable of generating a display and of transmitting operator responses to the IBM, but more complex operations, e.g. checking the syntax of an operator response, would require a USER program. EXEC will consist of three main parts: a monitor, an interrupt servicing routine, and a set of routines.

1. Monitor

The monitor will be a short program which will check certain software flags and will carry out actions requested by the IBM or the keyboard. The monitor will be in subroutine form so that the USER program can employ it if required, and when no USER program is running, the monitor will loop continuously. This arrangement ensures that although the IBM and the keyboard are serviced at interrupt level, the display update routines are called only from mainline level.

2. Interrupt servicing routine

There is only one interrupt level and therefore only one interrupt servicing routine, which handles the following devices:

a) AIP (IBM Accept Input interrupt)
Reads in a file from the IBM and acts according to the control function bits at the head of the file (see Appendix)

b) 33 $^{1}/3$ Hz synchronization
Starts next frame of display.
Checks keyboard. If set, reads character. If recognized control character, acts accordingly (see Appendix). If unrecognized control character, escapes to USER. If not control character, converts to EBCDIC, sets flag and returns to mainline.
Moves cursor, if required by relevant SKF.

c) Any other interrupt - escape to USER.

3. Subroutines

Subroutines do not preserve the accumulator, which is often used to pass parameters across. Subroutines are non-reentrant and must be called from mainline level only.

A provisional list of subroutines is:

ERMON	Error subroutine
INSCH DELCH	Single character insert and delete
INSST DELLN	Multiple character insert and delete (uses packed EBCDIC)

DRXYL DELXY }	Draw and delete horizontal or vertical line
DRGRA	Draw graph (see below)
DELGR	Delete graph
CLSCR	Clear screen (except system area)
TRFIL	Transmit a file to IBM (USER supplies control function, word count).

DRGRA is rather complex. For book-keeping purposes, each set of axes and each graph on that set of axes will be numbered. The following parameters are required for each set of axes:

a) book-keeping number

b) origin, given as a position on the character-matrix

c) length of axes, given in character-matrix units

d) range of x and y values, i.e. values to be assigned to extremities of axes

e) number of subdivisions of each axis

f) labels for each axis.

For each graph we require:

a) book-keeping number

b) histogram or point plot

c) list of pairs of (x,y) values.

This standard version of DRGRA will be placed at the end of EXEC. If a user requires a version capable of drawing continuous curves (which would extend beyond the 1 K boundary of EXEC), he can overlay the end of EXEC with a program containing his own DRGRA. In this case the USER program must exit by forcing a cold start rather than by returning to the monitor loop.

General procedures

Initially a program request will cause a program to be queued to the partition in the normal way. A standard program request unit will be mounted near each PDS1 display as they may act as back-up units for each other. Procedures after this will be chosen for each application, so as to compromise between (e.g.) operator waiting time and partition usage. Some examples are shown in Fig. 2 (PCL = PDS1 coreload, ICL = IBM partition coreload).

Distribution

CO Computer Section
E. Asséo
S. Battisti
G. Baribaud
G. Benincasa
M. Bouthéon
C. Bovet
G. Brianti
G. Daems
A. Daneels
D. Dekkers
P. Heymans
F. Hofmeister
Frank James
H. Koziol
P. Lefèvre
J.H.B. Madsen
A. Millich
K.O.H. Pedersen
K.H. Reich
Ch. Serre
G. Shering
K. Schindl
P.H. Standley
U. Tallgren

APPENDIX

Control functions for files from IBM to PDS1 (8 bits)

00000000	Cold start program
0xxxxxxx	Undefined (escape to USER)
10	Binary file follows:
10xxxx	Load into core, set SKF and return to mainline
11xxxx	Load into core and enter
0xxxxx	Load into core and wait for chained file
11	EBCDIC or Graphical file follows:
0xxxxx	EBCDIC
1xxxxx	Graphical
x0xxxx	Load into buffer, set SKF and return to mainline
x10xxx	Display (via monitor) and return to mainline
x11xxx	Display (via monitor) and return to mainline. Set "return changes made". Monitor will store changes made from keyboard and transmit them to the IBM when TRANSMIT key is pressed

Control functions for files from PDS1 to IBM (8 bits)

00000000	Not allowed
0xxxxxxx	Undefined (passed to partition)
100xxxxx	Abort
101xxxxx	Queue as per queue indicator
11	
00xxxx	Changed characters file
01xxxx	EBCDIC
10xxxx	STARC format
11xxxx	Section of display list (for saving a reference curve)

Control key allocations

↑↓→←	Initiate cursor movement
CONT 5	Rapid cursor movement (normally single shot)
CONT 6	Stop cursor movement
HOME	Cursor home
CR or LF	New line (end of statement)
TAB	Skip to 2nd column of display
DEL	Delete, cancel, ignore. Converted to -1
TRANSMIT	Send changes to IBM

Two-key allocations

↑A	Abort program, transmit abort request to IBM, return to EXEC monitor
↑B	Change to back-up keyboard.

This leaves 8 control keys and about 30 two-key combinations free for user allocation.

TIME →

<table>
<tr><td rowspan="5">P R O G R A M
R E Q U E S T</td><td rowspan="5">ICL 1
queued to position</td><td rowspan="5">Wait</td><td>ICL 1 loads PCL 1 into PDS 1 exits</td><td>PCL 1 conducts discussion (partition free)</td><td>PCL 1 requests ICL n to be queued (n depends on discussion</td><td>Wait</td><td>ICL n communicates with PCL 1</td><td rowspan="2">PCL 1 exits
ICL computes
(PDS 1 free)</td><td rowspan="2">ICL sends display data to PDS 1 EXEC and exits</td><td rowspan="2">Data displayed
(Partition & PDS 1 free)</td></tr>
<tr><td>ICL 1 loads PCL 1, sets SKF 3 ICL 2, exits</td><td>PCL 1 conducts discussion (partition free)</td><td>PCL 1 sends PRQ</td><td>Wait</td><td>ICL 2 communicates with PCL 1</td></tr>
<tr><td>ICL 1 loads PCL 1, starts computing</td><td>PCL 1 & ICL 1 compute, communicate, display, discuss</td><td>PCL 1 & ICL 1 exit (partition & PDS 1 free, display not destroyed)</td></tr>
<tr><td rowspan="2">ICL 1 computes</td><td>ICL 1 loads PCL 1 into PDS 1</td><td>ICL 1 & PCL 1 communicate</td><td>PCL 1 generates display (partition free)</td><td>PCL 1 exits (PDS free, display not destroyed)</td></tr>
<tr><td>ICL 1 sends display data to PDS 1 EXEC and exits</td><td>Data displayed (partition and PDS 1 free)</td></tr>
</table>

(Machine cycle or operator request)

Fig.2 Examples of program sequences